AMENDMENTS TO THE CLAIMS:

1. (Previously presented) A method of increasing at least one of efficiency and speed in executing a matrix subroutine on a computer, said method comprising:

storing data contiguously for a matrix subroutine call in a computer memory in an increment block size that is based on a cache size of said computer, wherein said cache comprises a cache having a cache size CS and said block increment size comprises a block of size 2NB by NB/2, wherein $NB^2 = \alpha$ CS, $\alpha < 1$, so that said block occupies a sizeable portion of said cache, said contiguous blocks of data thereby increasing efficiency and/or speed by providing data that will remain resident in said cache and is arranged in an order expected for processing of said matrix subroutine call.

2. (Rejected) The method of claim 1, further comprising:

retrieving said data from said memory in units of said increment block; and

executing at least one matrix subroutine using said data.

3-4. (Canceled)

5. (Rejected) The method of claim 1, wherein said cache comprises an L1 or L2 cache, said L1 or L2 cache comprising a cache closest to at least one of a Central Processing Unit (CPU) and a Floating-point Processing Unit (FPU) of a computer system associated with said computer memory.

6. (Rejected) The method of claim 1, wherein said matrix data is loaded contiguously in said memory in increments of a memory line size LS and data is retrievable from said memory in units of LS.

7. (Rejected) The method of claim 2, wherein said at least one matrix subroutine comprises a matrix operation.

8. (Rejected) The method of claim 2, wherein said at least one matrix subroutine comprises a subroutine from a LAPACK (Linear Algebra PACKage).

9. (Rejected) The method of claim 2, wherein said subroutine operates on an increment block of data as a result of a single call on this data.

10. (Previously presented) An apparatus, comprising:

a processor for processing a matrix subroutine;

a cache associated with said processor; and

a memory, wherein said memory stores data for memory calls of said matrix subroutine as contiguous data in an increment block size that is based on a dimension of said cache and loads said blocks of data into said cache for said matrix subroutine processing, wherein said cache comprises a cache having a cache size CS, and said block increment size comprises a block of size 2NB by NB/2, wherein $NB^2 = \alpha$ CS, $\alpha < 1$, so that said block occupies a sizeable portion of said cache, said contiguous blocks of data thereby increasing efficiency and/or speed by providing data that will remain resident in said cache and is arranged in an order expected for

processing of said matrix subroutine call.

11. (Canceled)

12. (Rejected) The apparatus of claim 10, wherein said matrix subroutine comprises a matrix operation.

13. (Rejected) The apparatus of claim 10, wherein said matrix subroutine comprises a subroutine from a LAPACK (Linear Algebra PACKage).

14. (Rejected) The apparatus of claim 10, wherein a line size of said memory is LS and data is retrieved from said memory in units of LS, each said block of data being retrieved by usually an integral number of memory line retrievals.

15. (Previously presented) A computer program product for use with a computer, said computer program product comprising a machine-readable medium tangibly embodying a program of machine-readable instructions executable by a digital processing apparatus, said instructions including a method of storing data for a matrix subroutine call in a computer memory in an increment block size that is based on a cache size of said computer, wherein said cache comprises a cache having a cache size CS, and said block increment size comprises a block of size 2NB by NB/2, wherein $NB^2 = \alpha\, CS$, $\alpha < 1$, so that said block occupies a sizeable portion of said cache, said contiguous blocks of data thereby increasing efficiency and/or speed by providing data that will remain resident in said cache and is arranged in an order expected for

processing of said matrix subroutine call.

16. (Rejected) The computer program product of claim 15, wherein said matrix subroutine comprises a subroutine from a LAPACK (Linear Algebra PACKage).

17. (Canceled)

18. (Rejected) The computer program product of claim 15, wherein a line size of said memory is LS and data is retrieved from said memory in units of LS, each said block of data being retrieved by usually an integral number of memory line retrievals.

19. (Previously presented) A method of solving a problem using linear algebra, said method comprising at least one of:

initiating a computerized method of performing one or more matrix subroutines, wherein said computerized method comprises storing data for a matrix subroutine call in a computer memory in an increment block size that is based on a cache size of said computer, wherein said cache comprises a cache having a size CS, and said block increment size comprises a block of size 2NB by NB/2, wherein $NB^2 = \alpha\, CS$, $\alpha < 1$, so that said block occupies a sizeable portion of said cache, said contiguous blocks of data thereby increasing efficiency and/or speed by providing data that will remain resident in said cache and is arranged in an order expected for processing of said matrix subroutine call;

transmitting a report from said computerized method via at least one of an internet interconnection and a hard copy; and

receiving a report from said computerized method.

20. (Canceled)

21. (Rejected) A method of providing a service, said method comprising an execution of a matrix subroutine in accordance with the method of claim 1.

22. (Currently amended) A method of providing a service, said method comprising ~~at least one of~~:

solving of a problem using linear algebra in accordance with the method of claim 19; and

providing a consultation to solve a problem that utilizes said computerized method.

23-24. (Canceled)

25. (Previously presented) The method of claim 1, further comprising:

processing said rectangular blocks of matrix data by calling a DGEMM (Double-precision Generalized Matrix Multiply) kernel a plurality of times, using each one of said rectangular blocks of contiguous data with each said DGEMM kernel call.

26. (Rejected) The method of claim 25, wherein data for all operands used in said DGEMM kernel comprise data stored as contiguous data in lines of said memory such that data for each said operand can be retrieved as contiguous data from said memory using usually an integral number of memory line retrievals respectively appropriate for each said operand.

27. (Previously presented) The method of claim 1, said method further comprising:

for data of said matrix, preliminarily converting and storing in said memory said data of said matrix into a number of rectangular blocks of contiguous data that will each fit into said cache size approximately NB x NB, including, if necessary, adding padding data to fill up a block or a complete line of memory, said padding chosen to have no effect on a calculation result of said matrix subroutine call.